Optimization of Sole Production Scheduling

Ana Moura [1], Cristina Lopes [2]†, João Matias [3], Nuno Lopes [4], and Ricardo Enguiça [4]

[1] *LEMA-ISEP, Polytechnic of Porto, and CMUP, University of Porto*
[2] *CEOS.PP, ISCAP, Polytechnic of Porto*
[3] *CMAT-UTAD - Centre of Mathematics*
[4] *ISEL, Polytechnic of Lisboa, and CEMAT, University of Lisboa*

(*Communicated to* MIIR *on 14 March 2024*)

Study Group: European Study Group with Industry 174, Portugal, 3rd-7th July 2023.
Communicated by: ESTG, Polytechnic of Porto.
Industrial Partner: Softideia, https://www.softideia.com/.
Presenter: António Macedo.
Team Members: Ana Moura, Cristina Lopes, João Matias, Nuno Lopes, Ricardo Enguiça.
Industrial Sector: Computing/Robotics; Manufacturing; Textile/Clothing/ Footwear.

Key Words: Manufacturing, Molds, Rubber soles, Scheduling, Industrial Engineering, Operational Research, Mathematical Modeling, Optimization, Heuristics.

MSC2020 Codes: 90B22, 90C05, 90C11.

Summary

This report addresses a challenge proposed at the 174th European Study Group in Industry to develop an algorithm to optimize processes on rubber soles manufacturing. The objective is to create an order distribution plan that maximizes efficiency, reduces setup times synchronizing molds changing, ensuring that delivery dates are met. This problem can be classified as a parallel machines scheduling problem with sequence dependent setup times. This work presents an heuristic that minimizes the makespan, accounting for the sequence dependent setup times, divided into two steps: (i) a client's order will be partitioned into sets corresponding to the assigned slots; (ii) reorder the molds within each position of the press, so that setup times are simultaneous. We find solutions for real data, also for cases with double and triple molds for a given size. Results show that synchronization of setup times due to heating and mold changes decrease significantly the makespan.

† Corresponding Author: `cristinalopes@iscap.ipp.pt`

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X Content not peer-reviewed by MIIR.

1 Introduction

At the 174th European Study Group in Industry, a challenge was proposed by the software company *Softideia* to develop an algorithm to optimize processes on rubber soles manufacturing.

The challenge consists in the optimization of the scheduling processes on a company that manufactures rubber soles, which involves the efficient management and distribution of its production orders by a set of available machines. The machines are divided into two sectors: Injection and Compression. Each machine works with one or more presses, where each press can operate with a specific number of molds. Each mold, in turn, produces a pair of soles in each production cycle.

The main question is how to distribute efficiently and effectively the production orders, taking into account the required quantities and sizes, across the available machines, considering several assumptions. The objective is, therefore, to create a production order distribution plan that maximizes efficiency, reduces the time for configuration and change of molds and colors, and ensures that delivery dates are met.

1.1 Scheduling problems

Scheduling problems are a daily challenge in manufacturing. In injection molding industry it is necessary to plan in advance the production scheduling, to maximize the system productivity and resource efficiency [5]. In some environments there are several machines that can be used in parallel to process jobs [7], and usually the objective is to optimize a certain time-related metric [9].

For each job j in a schedule, there are several metrics regarding the time spent processing the tasks in the system. The *completion time* of job j is usually denoted by C_j. The *makespan* is the completion time of the last job to leave the system:

$$C_{max} = \max_j C_j.$$

Minimizing the makespan usually implies a good use of machines [11] and therefore it is one of the most common performance measures.

Setup time is the time required to prepare all necessary resources to perform a task [1]. Setup times can be sequence dependent [6], machine dependent [8], or both: s_{ijk} represents the setup time on machine i between the processing of jobs j and k. Since the setup process does not add value, considering it explicitly in the scheduling decisions can improve resource utilization and increase productivity. Ignoring setup times adversely affects the solution quality of some applications of scheduling [1]. The classical review papers on scheduling problems with setup times by Allaverdi [2, 3, 1] discuss thoroughly many types of scheduling problems addressed in literature throughout the years.

In this work, minimizing the makespan is the main objective, since finishing the order of a client as soon as possible helps in meeting the established deadlines, decreasing idle times, not mixing up orders from different customers, and making the process efficient. The considered problem can be classified as a *parallel machines scheduling problem with sequence dependent setup times*, since there are 39 slots available that can be used in

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X Content not peer-reviewed by MIIR.

parallel, some jobs can only be processed in a subset of the available slots, and setup times are dependent on the sequence of jobs assigned to each machine.

1.2 Methodology

Scheduling problems can be solved either by using an exact approach, or an approximate approach. The exact approach means formulating a linear programming model and solving it with commercial solvers until finding the solution with the best possible value for the defined objective function (the *optimal* solution). The approximate approach means using an heuristic to build a feasible solution and improve it from there. The exact method finds the optimal solution of the problem, but if the size of the instance is large, it may take a considerable amount of time to find it, or even to prove that it is actually the best solution. The approximate method is usually fast in finding a good solution for the problem, but it does not guarantee its optimality.

Regarding approximate methods, there are specifically developed heuristics to target some performance measures of scheduling problems, known as dispatching rules. Dispatching rules are simple heuristic procedures, specially designed to generate a feasible solution, that perform well in a certain scheduling problem. More general heuristic procedures, such as simulated annealing, tabu search [13], or metaheuristics [10], can be used for solving scheduling problems as well.

The development of heuristic methods dedicated to address industrial needs, such as the ones presented in the next sections, can provide a fast, easy to build, and easy to maintain tool, which gives reasonable solutions over all objectives [14]. As noted in [12], allowing for the possibility of splitting jobs into inconsistent batches can improve the operational efficiency in parallel machine scheduling. In the proposed heuristic for this report, job splitting was considered for the cases where double or triple molds were available, aiming to synchronize the setup times.

1.3 The challenge for scheduling the rubber soles vulcanization process

In order to make strong and durable rubber, it goes through a heat-treatment phase known as vulcanization. This is where the rubber is cooked (often with sulphur) to create extra bonds or cross-links between the molecules of the rubber, so they don't easily fall apart [4]. Charles Goodyear accidentally discovered this process, when he dropped some rubber onto a hot stove and noticed how the heat made the rubber harder and more durable.

The optimization of the planning for the rubber soles vulcanization process is the focus of our report. To present the scheduling challenge, we will adopt the following terms.

- order - an order from a client for a certain sole model is a list with the quantities to be manufactured for each size.
- mold - there is a certain number of molds available for each size of a sole model; some sizes with larger demand may have multiple molds, that can be used simultaneously.
- slots - positions where a mold is put to compress the rubber to produce a pair of soles. There are slots of different sizes/types to fit mold types up to a certain height.

Figure 1. Molds used for manufacturing a pair of rubber soles.

Source: https://www.indiamart.com/proddetail/double-heel-type-shoe-sole-molding-die-5009140212.html

Source: https://www.tradewheel.com/p/rubber-sole-vulcanizing-machine-high-precision-468201/

- press - a press can have 2, 3, 4 or 5 slots of the same height. It compresses and heats up all the slots simultaneously, causing the vulcanization of the rubber.
- machine - a machine can have 2, 3 or 4 presses, being usually assigned to the same worker.

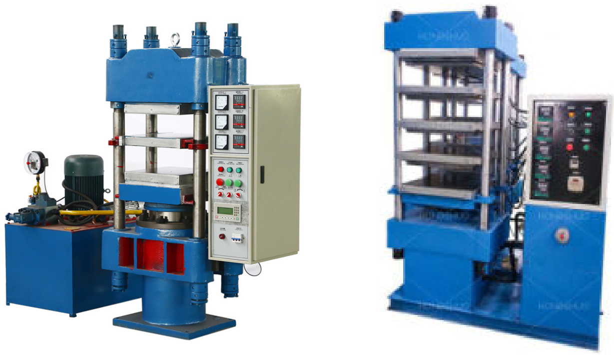

Figure 2. Vulcanization machines with one or more presses, each composed by several slots for positioning the molds.

Source: https://portuguese.alibaba.com/p-detail/Rubber-60625300797.html?spm=a2700.details.0.0.7f3646feRS6m0h

Source: https://xchsjx.en.made-in-china.com/product/gdCtpou0ArRM/China-Rubber-Bricks-Making-Machine-Rubber-Brick-Vulcanizing-Press-Gym-Floor-Tile-Making-Machine.html

Currently, the company has five machines available - A, B, C, D, and E. Machines A and B have 2 presses each, machines C and D have 3 presses, and machine E has 4 presses. The two presses of machine A have, respectively, 5 and 4 slots. Both presses in machine B have 4 slots. One of the presses in machine C has 3 slots and the other two have 2 slots. Machine D has the same configuration of machine C. All the four presses in machine E have 2 slots. This amounts to a total of 5 machines, with 14 presses, and 39 slots available to produce rubber soles simultaneously.

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X

Figure 3. Vulcanization machines with one or more presses, each composed by several slots for positioning the molds.

Source: https://www.maquinasmora.com.br/prensasvulcaniza%C3%A7%C3%A3o

Machines	A	B	C	D	E
#Presses	2	2	3	3	4
#Slots	5-4	4-4	3-2-2	3-2-2	2-2-2-2

Presses that have 5 slots can only fit thinner molds, while presses with 2 slots can fit thicker molds. When a certain client's order arrives, it is classified according to the height of the slots where the corresponding molds may fit. Therefore, we will consider four order types: T2, T3, T4, and T5. The most thin soles are T5 and the most thick ones are T2.

A more detailed version of order types will be given later, concerning also the number of multiple molds and the number of positions used to perform the order.

The mean time for the vulcanization process of a pair of soles is 10 minutes. We will consider this cycle time as the main timestep. The average downtime due to heating after changing a mold in one of the slots of a press is 30 minutes (3 timesteps). The replacement of a mold in one of the slots affects all slots of that press. During the heating time, when a new mold is inserted in a slot, all slots of the same press must also stand by, to adjust the temperature before continuing production. Therefore, to solve this scheduling problem, one must consider setup times dependent on the sequence in which jobs are assigned to each machine.

2 Heuristic

In this section, we present an heuristic that aims to find a solution to the problem in a fast way, and that highly satisfies the conditions desired by the company. By *solution*, we mean a distribution of a client's order by the slots of a press of an appropriate vulcanization machine that will be completed before the deadline accorded with the client. The following conditions were considered constraints in our problem:

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X

(1) a customer's order is manufactured in a single take, i.e., it does not alternate with other orders;

(2) within an order, all pairs of a given mold are manufactured in one take on the same press;

(3) if a sole size has more than one mold associated, then the soles of this size can be divided in any possible way;

(4) all pairs of a given customer's order are of the same height type;

(5) any pair of soles takes one timestep to be vulcanized;

(6) a mold takes three timesteps for heating before starting the process of vulcanization, and all slots in the same press stop working in that three time steps.

The following table summarizes a real instance of the problem, provided by the company.

Size	36	37	38-1	38-2	39-1	39-2	40	41	42	43
Nb. of Molds	1	1	2		2		1	1	1	1
Order Quantity	212	460	964		1234		839	339	144	96
Quantity per mold	212	460	482	482	617	617	839	339	144	96
Order (**O**)	215	463	970		1240		842	342	147	99

Table 1: Example of a client's order provided by the company.

Given a certain client's order, let n be the number of different soles sizes, and l be the total number of molds. For the example in Table 1 we have $n = 8$ and $l = 10$. In the table, the Order Quantity for each size sole is given in the third line, where we can see that there are two molds for each of the sizes 38 and 39. In the fourth line, the company presented the standard split of the order quantities per mold. Since each mold needs three time steps for heating before start the vulcanization, and takes one time step producing one pair of soles, we added those three times steps per mold to the order quantity, resulting in the values presented in the fifth line of the table. We set $\mathbf{O}$ to be this vector, $\mathbf{O} = (215, 463, 970, 1240, 842, 342, 147, 99)$. This order will be produced in a single machine that has a certain number of presses of the height type of the order, in a total of k slots.

We label the order according to the mold height, and the number of double or triple molds, using the notation $M_{\_}(S_{\_}D_{\_}T_{\_}, k)$, where the subscribed values will correspond, respectively, to the mold height type, and the number of single, double and triple molds, and k is the total number of positions in the machine where we could vulcanize the considered type of sole. For the example of Table 1, considering that the mold is of height type 4, and that will be vulcanized in a machine with two presses of this height type, we denote the order by $M_4(S_6D_2, 8)$. Another example, $M_2(S_5D_2T_1, 8)$ would correspond to an order of molds of height 2, with 5 single molds, 2 double molds and 1 triple mold, making a total of 12 available molds, and that will be done in a machine with 4 presses of height type 2, in a total of 8 positions for vulcanization. Note that n is the sum of the subscribed values in S, D, and T, and l is the sum of these values multiplied, respectively, by 1, 2, and 3 (the number of molds of each size).

Given a client's order, a certain machine will be chosen to vulcanize all the soles of the order, and that could be busy with a previous order of another client. Thus, we need to consider a starting times vector for all the positions in the machine to be used by the order. We set $\mathbf{B}$ to be the vector of availabilities of each position, with length k. To run the heuristic for a single client, we set the first starting time to be zero.

Given an order $\mathbf{O}$, of length n, to be produced with l molds, in a machine with k positions, we define a word W of length l in k letters, where $W_i \in \{1, \ldots, k\}$ will be the slot assigned to mold of size i where the vulcanization will be done.

To calculate the instant when a slot in a press will finish the vulcanization of its assigned soles, we need to add five types of time intervals: (i) the time that the slot is still working on the previous client's order; (ii) the time to vulcanize soles assigned to the slot; (iii) three time steps whenever the press needs to heat a new mold in that slot (heating); (iv) the intervals of time when the slot will be empty, waiting for the heating of a new mold in another slot in the same press (waiting); (v) the intervals of time when the slot will be empty, waiting to synchronize with another slot for heating new molds (synchronization). Our goal is to minimize the completion time of the order.

The heuristic consists in splitting the problem into two distinct phases, making an exhaustive search within each sub-problem. Step (i) sets which molds go to each position, Step (ii) finds the ordering of the molds within each position that maximizes the number of simultaneous changes of molds.

This procedure works well and finds practical solutions for this scheduling problem.

In the following subsections we will explain the details of Steps (i) and (ii), and we will provide several examples in Section 3, namely the cases with only single molds, one double mold, one triple mold, and one example with a double and a triple mold.

2.1 Step (i): The k-partition of a client's order

In the first step of the heuristic, a client's order will be partitioned into sets corresponding to the available slots in the machines. There are some variations in the heuristic when the order allows for some sizes with double or triple molds.

2.1.1 *All single molds*

We start by assuming that all molds are single, that is $n = l$. Considering k slots to vulcanize the corresponding soles, all possible combinations of size soles and press positions will be analyzed. This corresponds to evaluate all words W of length l in k letters, where the i^{th} letter W_i will represent the slot where the vulcanization of all soles of mold i will be produced. Each word W is called a k-partition of the order. For example, letting $l = n = 3$, the word $W = [2\,2\,1]$ corresponds to an order of three soles sizes, where the first two are assigned to slot 2, and the third is assigned to slot 1.

We set $\mathbf{E}_W = (E_j)_{j=1,\ldots,k}$ to be the vector of the ending times of the slots used to

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X

produced the given order. It follows that

$$E_j = B_j + \sum_{i \in I_j} O_i,$$

where I_j, $j = 1, \ldots, k$, is the set of positions i of W such that $W_i = j$. In other words, I_j is the subset of the soles of the order that are assigned to slot j. Note that these ending times do not consider yet the waiting and synchronization times, as they will be considered in Step (ii).

In Step (i), our goal is to choose W such that $||\mathbf{E}_W||_\infty = \max_j E_j$ is minimized,

$$\min_W ||\mathbf{E}_W||_\infty,$$

i.e., we seek to minimize the *makespan*, the time instant in which the vulcanization of all order ends. The chosen W is associated with a minimal k-partition $\mathcal{P}_W$ of the order.

In case we have at least one sole size with more than a mold, i.e. $l > n$, we divide this size order of length n in l blocks, before the calculation of the minimal k-partition. In the following subsections, we define a procedure, in the attempt to achieve the best solution, the one that minimizes the finishing time of all order.

2.1.2 *One double mold*

For the case with one double mold, that is, where we can use two molds to produce the soles of a certain size in a given order, an additional reasoning has to be introduced. In this case, for an order $\mathbf{O}$ of size n, we have $n-1$ single molds and 2 molds corresponding to the doubled number. So we will consider two order vectors, one for the quantities of sole sizes with single molds, $\mathbf{O_1}$, and another for the soles with double mold, $\mathbf{O_2}$, where we divide the total quantity by the two molds.

We define the ratio r to be

$$r = \frac{\sum \mathbf{O_1} + \sum \mathbf{O_2} + \sum \mathbf{B}}{k}.$$

Note that if we were able to obtain an ideal configuration, r would be the smallest completion time possible.

As in the previous case, we will consider the set of k-partitions of $\mathbf{O_1} + \mathbf{O_2}$, and find the partition that minimizes a balanced metric for the two objectives:

$$\min \left[10 \sum_{i=3}^{k} \Big| E[i] - r \Big| \right] + \left[\Big| E[1] - E[2] \Big| \right].$$

The objective function is composed of two parts. The first one balances the $k-2$ last positions as close as possible to the ratio value r. The factor 10 makes it more important to make the associated terms of the sum closer to 0 than the last term. The last term of the objective function represents the difference between the completion times of the first two slots, where the double molds are assigned. The quantity of the size order for these molds will be divided in such a way that the completion times of two slots are as close to each other as possible.

Since the mold of $\mathbf{O_2}$ can be produced on both positions 1 and 2, we will split the total amount of the order of the double mold in a way that, adding the split values to $E[1]$ and $E[2]$, they end as close as possible (and as a consequence of the objective function, also close to r).

2.1.3 *One triple mold*

The reasoning for this case will just consist on a simple adaptation of the one considered for a double mold, defining instead of $\mathbf{O_2}$ a vector $\mathbf{O_3}$, and letting 3 positions out to perform the production of the number associated to the tripled mold. The corresponding metric to minimize will be

$$\min\left[10\sum_{i=4}^{k}(|E[i]-r|)\right]+\left[\Big|E[1]-E[2]\Big|+\Big|E[2]-E[3]\Big|+\Big|E[3]-E[1]\Big|\right],$$

with

$$r=\frac{\sum\mathbf{O_1}+\sum\mathbf{O_3}+\sum\mathbf{B}}{k}.$$

As in the last subsection, the objective function is composed of two parts. The first one balances the $k-3$ last positions as close as possible to the ratio value r. The last terms of the objective function represent the difference between the completion times of the first three slots, where the triple molds are assigned. The quantity of the size order for these molds will be divided in such a way that the completion times of the three slots are as close to each other as possible.

2.1.4 *Two double molds*

For this case we assume that the number of positions is at least 4, since otherwise we could not take full advantage of the split of two molds. As in the previous cases, we consider

$$r=\frac{\sum\mathbf{O_1}+\sum\mathbf{O_2}+\sum\mathbf{B}}{k},$$

and also define d as the difference between the two values of the two sizes with double molds, that will be divided in two values each for vector $\mathbf{O_2}$, with length 4. In this case, in the metric we will consider two extra terms that balance the four positions used for the double molds in a way that the difference between the two pairs is as close to d as possible. The metric formula is the following:

$$\min\left[10\sum_{i=5}^{k}\Big(|E[i]-r|\Big)\right]+\left[\Big|\Big|(E[1]+E[2])-(E[3]+E[4])\Big|-d\Big|\right]+\left[\Big|E[1]-E[2]\Big|+\Big|E[3]-E[4]\Big|\right].$$

The first term aims to balance the $k-4$ positions (the ones with only single molds) as close to the ideal ratio r as possible. The second and third terms split the order produced by the double molds in such a way that the completion times of the four slots are as close to d and to each other as possible.

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X Content not peer-reviewed by MIIR.

2.1.5 *One double and one triple mold*

For the case where one double mold and one triple mold are available, we consider

$$r = \frac{\sum O_1 + \sum O_2 + \sum O_3 + \sum B}{k}\,,$$
$$r_2 = 2r - \sum O_2\,,\text{ and}$$
$$r_3 = 3r - \sum O_3\,.$$

In this case, in the metric we will consider an extra group of terms that balances the five positions used for the double and triple molds in a way that the difference between the values is as close to r_2 and r_3, respectively, as possible. The objective is to minimize the following formula:

$$\min\left[10\sum_{i=6}^{k}\Big(|E[i]-r|\Big)\right] + 5\left[\Big|E[1]+E[2]-r_2\Big| + \Big|E[3]+E[4]+E[5]-r_3\Big|\right] + \\ + \left[\Big|E[1]-E[2]\Big| + \Big|E[3]-E[4]\Big| + \Big|E[4]-E[5]\Big| + \Big|E[3]-E[5]\Big|\right].$$

The objective function is composed of three parts. The first one balances the $k-5$ last positions as close as possible to the ratio value r. The second group of terms of the objective function represents the difference between the completion times of the first five slots, where the double and triple molds are assigned. Note that r_2 represents the order quantity produced with single molds placed in positions 1 and 2 and r_3 represents the order quantity produced with single molds placed in positions 3, 4 and 5. The third group of terms divides the quantity of the size order for the double and triple molds in such a way that the completion times of five slots are as close as possible. The factors 10 and 5 give more relative importance to the first and second group of terms.

2.2 Step (ii): Ordering within a press

Let W be the word determined in Step (i) and $\mathcal{P}_W$ the associated partition of the order into k sets. Note that in Step (i) the quantities of $\mathbf{O_2}$ and $\mathbf{O_3}$ are already divided.

Since this is a scheduling problem with sequence dependent setup times, producing the rubber soles in a different sequence may change the final completion time. In this second step we evaluate all the permutations of the molds within each position in order to minimize the heating times in a press. This is accomplished by choosing an ordering of the molds in each slot that allows for the most coincidences in the change of molds, in order to avoid unnecessary delays.The result is an improved solution.

Note that if an order is produced in more than one press, we only have to synchronize mold changes for positions in the same press.

For each j, with $j = 1, \ldots, k$, let $\mathbf{C}_j$ be the vector of cumulative times at position j, called *timeline* from now on. This vector is the sum of the timesteps necessary to produce the soles with the molds assigned to position j, added with the corresponding heating time, and also with the waiting times caused by the heating in another slot in the same press. These cumulative times can be improved by promoting simultaneous mold changes.

For two positions $i \neq j$ in the same press, if there exist $c \in C_i$ and $c^* \in C_j$ such that $|c - c^*| < 3$, then the change of the associated molds will be synchronized at the latest time. In this case, the slot that finishes first will wait one or two timesteps to start the next mold simultaneously to the start of the mold in the other slot. The timeline is then updated with the delay in this position. For that reason, the analysis of synchronizable mold changes is carried out chronologically.

For time distances of more than 3 timesteps between cumulative times, we opted not to synchronize, but the algorithm could be easily adapted if it is considered worthy.

3 Examples

In this section we present several examples for a better understanding of the heuristic, which varies according to the case where all sizes have single molds or the cases where some sizes have double or triple molds.

3.1 All single molds

In this section we deal with the simplest case, where all soles sizes have one single mold. In order to better understand the procedure, let us consider the real data provided by the company, presented in Table 1, and make the assumption that we do not have double molds (see Table 2).

Size	36	37	38	39	40	41	42	43
Nb. of Molds	1	1	1	1	1	1	1	1
Order Quantity	212	460	964	1234	839	339	144	96

Table 2: A client's order provided by the company, assuming that all sizes have single molds.

Assuming that the order will be performed in an unique press with 4 slots, and all slots are available from the initial time instant, the two steps of the algorithm produced the following results.

3.1.1 *Step (i)*

The word W in $k = 4$ letters (corresponding to the number of slots) and length $l = 8$ (number of different size soles) that minimizes the objective function is $W = [2\,1\,4\,3\,2\,1\,1\,1]$. The associated minimal k-partition consists of

$$\mathcal{P}_W = \{S_1 = [463, 342, 147, 99],\ S_2 = [215, 842],\ S_3 = [1237],\ S_4 = [967]\},$$

and $\mathbf{E}_W = [1051, 1057, 1237, 967]$, with $||\mathbf{E}_W||_\infty = 1237$.

Note that for simple examples it is quite common for the minimum value to be attained for several k-partitions, that is, there might not exist an unique minimal configuration (this is quite evident in this concrete example, where we have a huge amount of soles to be produced for one size with no double mold available).

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X Content not peer-reviewed by MIIR.

In this step, we do not make a decision about the ordering of the molds assigned to each slot. Figure 4 shows what will happen when the molds are put in each slot by ascending ordering of size sole. *Setup times* are represented by the vertical red lines corresponding to heating times of a new mold, and by the vertical pink lines corresponding to waiting times, when the press is stopped to heat a new mold in another slot. In that case, we will have $||\mathbf{E}_W||_\infty = ||\mathbf{E}_W||_\infty + 3(l-4) = 1249$, i.e., the completion time increased three time steps for each mold that needed to be heated after the initial time instant (it corresponds to the worst-case scenario).

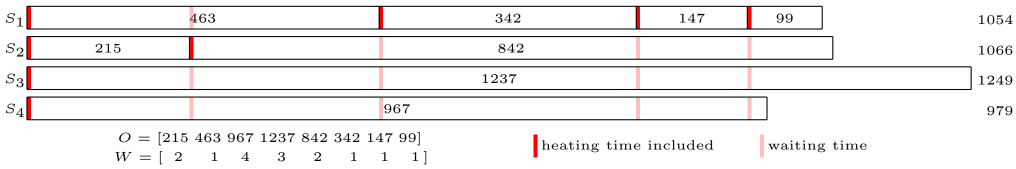

Figure 4. Step (i): the minimum 4-partition of the client's order, with the increase of waiting times in each slot, when the molds are put in ascending ordering of the size soles.

3.1.2 *Step (ii)*

In this step, we search for the optimal ordering of the molds in each slot, the ordering that minimizes the makespan, i.e., the completion time of the last job from the order. The 48 permutations were analyzed, and all are optimal solutions. In fact, in this case, Step (ii) did not improve the result obtained in Step (i), because it was not possible to synchronize the heating times of different slots. All the mold changes had a difference more than or equal to 3 timesteps.

Table 3 presents the scheduling of an optimal configuration, which corresponds to the solution presented in Figure 4.

Slot Id	2	1	1	1	4	1	2	3
Mold Id	36	37	41	42	38	43	40	39
Order	215	463	342	147	967	99	842	1237
Timeline	215	466	808	955	979	1054	1066	1249

Table 3: Scheduling of one of the 48 optimal permutations.

3.2 One double mold

In this section, we analyze an example where a double mold for a size will be available. Using the real data provided by the company, we make the following assumptions: two molds of size sole 39 are available (see Table 4), and the order will be performed in a unique press with 4 slots, which are all available from the initial time instant.

https://doi.org/10.33774/miir-2024-zhmg5 ORCID: https://orcid.org/0000-0002-4833-470X

Size	36	37	38	39	40	41	42	43
Nb. of Molds	1	1	1	2	1	1	1	1
Order Quantity	212	460	364	1234	339	339	144	96
Order (**O**)	215	463	367	1240	342	342	147	99

Table 4: Example of an order with one double mold.

Therefore, we have an order of type $M_4(S_7D_1, 4)$, where $\mathbf{B} = [0\,0\,0\,0]$, and

$$\mathbf{O_1} = [215\,463\,367\,342\,342\,147\,99] \text{ and } \mathbf{O_2} = [1240].$$

The results obtained by the two steps of the heuristic are the following.

3.2.1 *Step (i)*

The minimal k-partition obtained by the heuristic described in Section 2 is codified by the word $W = [4\,2\,1\,3\,4\,2\,1\,3\,1]$, and consists of

$$\mathcal{P}_W = \{S_1 = [367, 342, 99],\ S_2 = [463, 342],\ S_3 = [147, 654],\ S_4 = [215, 586]\},$$

with $\mathbf{E}_W = [808, 805, 147 + 654, 215 + 586]$, and $||\mathbf{E}_W||_\infty = 808$ (we highlighted with a different color the letters corresponding to the double mold, and a similar procedure will be made in the following examples).

Note that first, only single molds are assigned, and we get $E_1 = 808$ and $E_2 = 805$, which are close to $r = 803.75$, and $E_3 = 147$ close to $E_4 = 215$. After this, we split the value for $\mathbf{O_2}$ into 654 and 586, to be introduced respectively in E_3 and E_4, the slots where the double molds will be placed. Then, the waiting times are introduced, producing the final vector $\mathbf{E} = [817, 817, 813, 813]$. This is illustrated in Figure 5.

Figure 5 presents the scheduling for the vulcanization of the soles in the ascending ordering of size sole, and adding the associated waiting times. Note again that these waiting times were not taken into account in the computation of the minimal 4-partition.

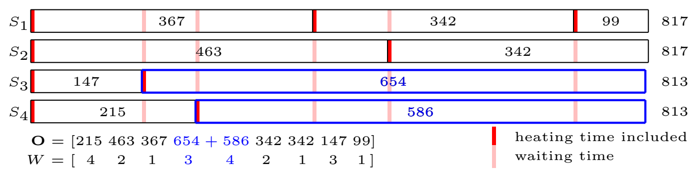

Figure 5. Step (i): the minimal 4-partition of the order, with one double mold, with the increase of waiting times in each slot, when the single molds are put in ascending ordering of size soles, and the double molds at the end.

3.2.2 *Step (ii)*

This procedure evaluates the possible synchronized mold changes.

After analyzing all 48 permutations in each slot for the molds ordering, the algorithm found 16 improved solutions. Table 5 and Figure 6 show one of these solutions.

Note that it was possible to synchronize the first mold change in slots S_1 and S_2.

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X

Step (ii) improved the solution from 817 to 814, with a decrease of 3 time steps in the completion time.

Slot Id	1	2	1	4	3	3	4	1	2
Mold Id	41	40	43	39_2	39_1	42	36	38	37
Order	342	342	99	586	654	147	215	367	463
Timeline	342	342	441	592	663	810	810	814	814

Table 5: Scheduling of one of the permutations.

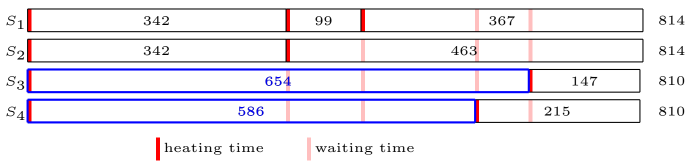

Figure 6. A solution for the scheduling of Table 5.

3.3 One triple mold

In this section, the heuristic is applied to an example with one triple mold based on the company's data. Once again, making a small change in the real data provided, we get a case with a triple mold in size 39 and all other sizes with single molds (see Table 6).

Size	36	37	38	39	40	41	42	43
Nb. of Molds	1	1	1	3	1	1	1	1
Order Quantity	212	460	364	1234	339	339	144	96
Order (**O**)	215	463	367	1243	342	342	147	99

Table 6: Example of an order with a triple mold.

This example is of an order of type $M_4(S_7T_1, 4)$, where $\mathbf{B} = [0\,0\,0\,0]$,

$$\mathbf{O_1} = [215\,463\,367\,342\,342\,147\,99] \text{ and } \mathbf{O_3} = [1243].$$

3.3.1 *Step (i)*

The minimal 4-partition obtained by the heuristic described in Section 2 is codified by the word $W = [3\,1\,4\,2\,3\,4\,2\,1\,3\,2]$, and consists of

$$\mathcal{P}_W = \{S_1 = [463, 342],\ S_2 = [342, 99, 364],\ S_3 = [215, 147, 442],\ S_4 = [367, 437]\},$$

with $\mathbf{E}_W = [805, 441 + 364, 362 + 442, 367 + 437]$, and $||\mathbf{E}_W||_\infty = 805$.

Figure 7 presents the scheduling for the vulcanization of the soles in the ascending ordering of size sole with single molds and with the double molds at the end, and adding the waiting times when a mold is heating in another slot.

https://doi.org/10.33774/miir-2024-zhmg5 ORCID: https://orcid.org/0000-0002-4833-470X Content not peer-reviewed by MIIR.

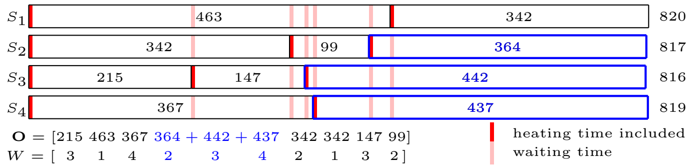

Figure 7. Step (i): the minimal 4-partition of the client's order, with one triple mold, with the increase of waiting times in each slot, when the molds are put in ascending ordering of the single size soles, and the triple molds at the end.

3.3.2 *Step (ii)*

Searching for the optimal permutations, we analyzed the 144 available permutations, and found 6 improved solutions. Table 7 and Figure 8 present one of the results.

Slot Id	1	2	2	4	3	3	3	2	1	4
Mold Id	41	40	43	39_3	39_2	42	36	39_1	37	38
Order	342	342	99	437	442	147	215	364	463	367
Timeline	342	342	441	441	448	595	810	811	814	814

Table 7: Scheduling of one of the 8 permutations.

Note that the three molds of size 39 have been placed in slots 2, 3 and 4. By reordering the molds in its slots, one can change molds simultaneously in slots 1 and 2 at instant 342. This achieved a saving of 3 timesteps in all slots. The next mold change happens at instant 440 at slot 4 and at instant 441 in slot 2. As the timeline difference between these two slots is smaller than 3 timesteps, then the heuristic makes slot 4 idle during one timestep to synchronize the mold change with slot 2 at instant 441 (Figure 8). By synchronizing these setup times, a saving of 3 timesteps can be achieved in all slots, except for slot 4 that saves only 2 timesteps.

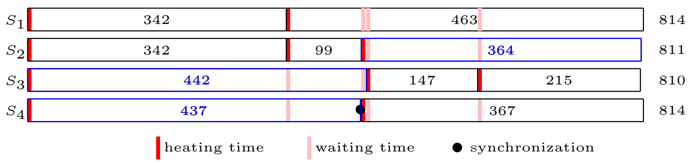

Figure 8. A solution for the scheduling of Table 7, for a client's order with one triple mold.

The next need to change molds is on slot 3 at instant 442+3+3=448 and then at instant 595. The algorithm was not able to schedule any setups in other slots at the same time as these. The makespan of this order is 814, which is 6 timesteps (which means 1 hour in only one client's order) better than the solution obtained in Step (i).

3.4 Two double molds

This case corresponds to the real data provided by the company (see Table 8), without any changes. We still assume that the order is made in a single press with 4 slots, consisting in an example of type $M_4(S_6D_2, 4)$, where $\mathbf{B} = [0\,0\,0\,0]$,

$$\mathbf{O_1} = [212\,460\,839\,339\,144\,96] \text{ and } \mathbf{O_2} = [964\,1234].$$

Size	36	37	38	39	40	41	42	43
Nb. of Molds	1	1	2	2	1	1	1	1
Order Quantity	212	460	964	1234	839	339	144	96
Order (**O**)	215	463	970	1240	842	342	147	99

Table 8: Example of an order with two double molds.

The minimal 4-partition obtained by the heuristic described in Section 2 is codified by the word $W = [3\,4\,1\,2\,3\,4\,2\,1\,3\,3]$, and consists of

$$\mathcal{P}_W = \{S_1 = [342, 735],\ S_2 = [842, 235],\ S_3 = [215, 147, 99, 621],\ S_4 = [463, 619]\}$$

with $\mathbf{E}_W = [342 + 735, 842 + 235, 461 + 621, 463 + 619]$, and $||\mathbf{E}_W||_\infty = 1082$.

Figure 9 presents the scheduling for the vulcanization of the soles in the ascending ordering of size sole with single mold and with double molds at the end, and adding the waiting times when a mold is heating in another slot.

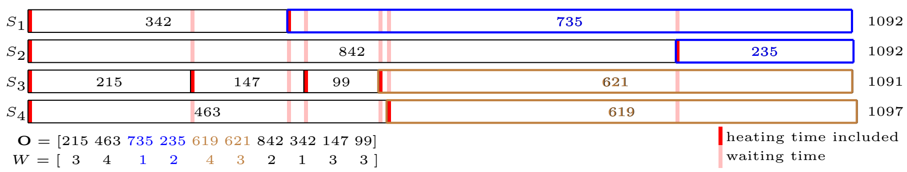

Figure 9. Step (i): minimal partition of the example with two double molds.

Searching for the permutations, we analyzed the 192 available permutations in Step (ii), and found 24 improved solutions. Table 9 and Figure 10 present one of the results.

Slot Id	2	3	4	3	1	3	3	1	2	4
Mold Id	38_2	39_1	39_2	43	38_1	42	36	41	40	37
Order	235	621	619	99	735	147	215	342	842	463
Timeline	235	624	624	723	744	873	1088	1089	1089	1096

Table 9: Scheduling of one of the 24 permutations.

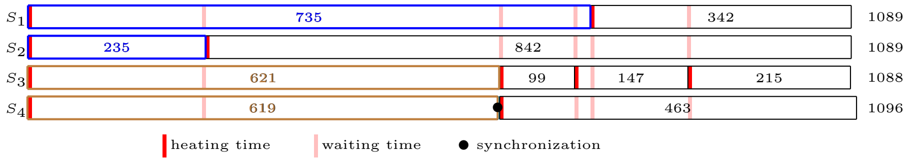

Figure 10. Step (ii): A solution with two double molds.

The first mold change happens at instant 235, but unfortunately any of the other slots can change molds at the same time. The next mold change would happen at 622 at slot 4 and at instant 624 at slot 3. Note that, since the difference of the timeline is smaller than 3 timesteps, slot 3 and slot 4 can synchronize the mold change at instant 624, if slot 4 is idle for 2 timesteps. This saves 3 timesteps in slots 1,2 and 3, and saves one timestep in slot 4. The heuristic improved one timestep from Step (i) to a final makespan of 1096 timesteps.

3.5 One double and one triple mold

Finally, let us consider an example with a double and a triple mold. We adapted the data provided by the company, and considered the order described in Table 10. We also suppose that the order will be performed in a unique press with 5 slots, which are all available from the initial time instant. This corresponds of an order of type $M_4(S_6D_1T_1, 5)$, where $\mathbf{B} = [0\,0\,0\,0\,0]$,

$$\mathbf{O_1} = [212\,460\,539\,339\,144\,96],\ \mathbf{O_2} = [964] \text{ and } \mathbf{O_3} = [1234].$$

Size	36	37	38	39	40	41	42	43
Nb. of Molds	1	1	2	3	1	1	1	1
Order Quantity	212	460	964	1234	539	339	144	96
Order (**O**)	215	463	970	1243	542	342	147	99

Table 10: Example of an order with a double and a triple molds.

For this case, the minimal 5-partition obtained by the heuristic is codified by the word $W = [3\,5\,1\,2\,3\,4\,5\,2\,4\,3\,1]$ and consists of

$$\mathcal{P}_W = \{S_1 = [99, 707],\ S_2 = [542, 263],\ S_3 = [215, 147, 442],\ S_4 = [342, 461],\ S_5 = [463, 340]\}$$

with $\mathbf{E}_W = [99 + 707, 542 + 263, 362 + 442, 342 + 461, 463 + 340]$.

Figure 11 presents the scheduling for the vulcanization of the soles, and adding the waiting times when a mold is heating in another slot. The solution obtained in Step (i) produces a makespan of 821 timesteps, which will be improved in Step (ii).

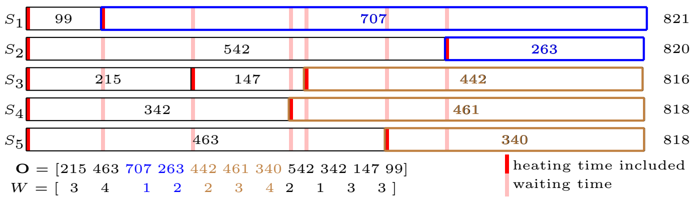

Figure 11. Step (i): minimal partition with a double and a triple molds.

In Step (ii), for this example, there are 96 available permutations of which 48 are improved solutions. One of these solutions is presented in Table 11 and in Figure 12.

Slot Id	2	4	5	3	3	1	3	4	2	5	1
Mold Id	38_2	41	39_3	39_1	42	38_1	36	39_2	40	37	43
Order	263	342	340	442	147	707	215	461	542	463	99
Timeline	263	345	345	448	595	719	813	815	817	817	818

Table 11: Scheduling of one of the 48 permutations.

The first mold change happens at instant 263, and it causes all other slots to wait 3 timesteps meanwhile. The next mold change would happen at instant 340+3=343 in slot 5 and at instant 342+3=345 in slot 4; since the difference is 2 timesteps, slot 5 is idle during this time difference so both slots can synchronize their mold changes. Due to this, 3 timesteps are saved in slots 1 to 4, and only one timestep in slot 5. None of the other mold changes could be synchronized. This step of the heuristic improved the solution from a makespan of 821 to 818 timesteps.

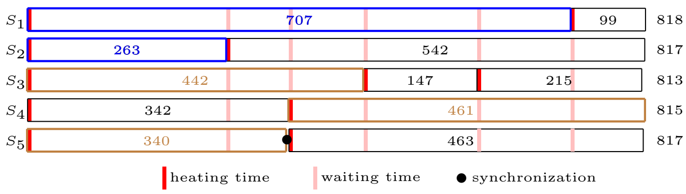

Figure 12. Step (ii): a solution with a double and a triple molds.

3.6 Summary of the results of the heuristic

The heuristic was run in several examples with data provided from the company. Table 12 provides a summary of the results obtained from the shown examples. All the computations were performed on a desktop computer running Linux Ubuntu 18.04, equipped with an AMD Ryzen 9 3950X processor and with 64Gb of RAM memory. The implementation of the heuristic was done using the **Julia** (v.1.9.0) language. In all cases, the computational times of running the heuristic were under three seconds.

https://doi.org/10.33774/miir-2024-zhmg5 **ORCID:** https://orcid.org/0000-0002-4833-470X Content not peer-reviewed by MIIR. **License:** CC BY-NC-ND 4.0

Instance	Type	n sizes	l molds	Step(i)	Step (ii) Makespan
All single molds	$M_4(S_8, 4)$	8	8	1249	1249
One double mold	$M_4(S_7D_1, 4)$	8	9	817	814
One triple mold	$M_4(S_7T_1, 4)$	8	10	820	814
Two double molds	$M_4(S_6D_2, 4)$	8	10	1097	1096
One double and one triple mold	$M_4(S_6D_1T_1, 5)$	8	11	821	818

Table 12: Summary of the results of the heuristic.

4 Conclusions and future works

This report presents an heuristic developed for the scheduling problem proposed by *Softideia* at the 174$^{\text{th}}$ ESGI. Based on a smart strategy, the algorithm finds a good solution in a short amount of time. The proposed methodology followed an heuristic, instead of an exact mathematical model, since for the company it is more important to find a good solution in a short amount of time, than spending possibly too much time running a computational model that would provide the exact optimal solution.

The time complexity of the proposed algorithm is not polynomial. In fact, the first step of the algorithm is exponential, and the second step is factorial. However, the instances that the company deal with are small, either regarding the number of different rubber soles sizes, or the 39 available slots. Therefore, the heuristic was able to find a good solution for all examples based on real data from the company in just a few seconds.

After choosing a machine, the corresponding smaller problem is solved to optimality with the proposed heuristic, since all options are being tested and compared. However, that solution is conditioned by the selection of the machine made on the first step and also by the strategy chosen for the double and triple molds. Hence, the final solution obtained by the heuristic is a good solution, but it is not proven to be the optimal solution of the initial framework.

In future research, the deadlines from the clients could be considered, making it possible to explore more sophisticated performance measures such as tardiness, lateness, or number of tardy jobs. An improved heuristic could also be developed, to explore more areas of the solution space than the explored ones, namely in the splitting jobs between double and triple molds, and in the first assignment of the machine type.

References

[1] A. Allahverdi. The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*, 246(2):345–378, Oct. 2015.

[2] A. Allahverdi, J. N. D. Gupta, and T. Aldowaisan. A review of scheduling research involving setup considerations. *Omega*, 27(2):219–239, Apr. 1999.

[3] A. Allahverdi, C. T. Ng, T. C. E. Cheng, and M. Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985–1032, June 2008.

[4] Coruba. UK manufacturer and distributor of quality assured rubber products, 2023.

[5] X. Fu, F. T. S. Chan, B. Niu, N. S. H. Chung, and T. Qu. A three-level particle swarm optimization with variable neighbourhood search algorithm for the production scheduling problem with mould maintenance. *Swarm and Evolutionary Computation*, 50:100572, Nov. 2019.

[6] D. Kress, D. Müller, and J. Nossack. A worker constrained flexible job shop scheduling problem with sequence-dependent setup times. *OR Spectrum*, 41(1):179–217, Mar. 2019.

[7] D. Lei and H. Yang. Scheduling unrelated parallel machines with preventive maintenance and setup time: Multi-sub-colony artificial bee colony. *Applied Soft Computing*, 125:109154, Aug. 2022.

[8] G. Li, M. Liu, S. P. Sethi, and D. Xu. Parallel-machine scheduling with machine-dependent maintenance periodic recycles. *International Journal of Production Economics*, 186:1–7, Apr. 2017.

[9] S. Maecker, L. Shen, and L. Mönch. Unrelated parallel machine scheduling with eligibility constraints and delivery times to minimize total weighted tardiness. *Computers & Operations Research*, 149:105999, Jan. 2023.

[10] M. Mousakhani. Sequence-dependent setup time flexible job shop scheduling problem to minimise total tardiness. *International Journal of Production Research*, 51(12):3476–3487, 2013.

[11] M. L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer US, Boston, MA, 2012.

[12] O. Shahvari and R. Logendran. Hybrid flow shop batching and scheduling with a bi-criteria objective. *International Journal of Production Economics*, 179:239–258, Sept. 2016.

[13] L. Shen, J. N. D. Gupta, and U. Buscher. Flow shop batching and scheduling with sequence-dependent setup times. *Journal of Scheduling*, 17(4):353–370, Aug. 2014.

[14] N. Srinath, I. O. Yilmazlar, M. E. Kurz, and K. Taaffe. Hybrid multi-objective evolutionary meta-heuristics for a parallel machine scheduling problem with setup times and preferences. *Computers & Industrial Engineering*, 185:109675, Nov. 2023.